Computer Science • 14 (3) 2013 http://dx.doi.org/10.7494/csci.2013.14.3.367

Krzysztof Wróbel
Paweł Torba
Maciej Paszyński
Aleksander Byrski

EVOLUTIONARY MULTI-AGENT COMPUTING IN INVERSE PROBLEMS

Abstract *This paper tackles the application of evolutionary multi-agent computing to solve inverse problems. High costs of fitness function call become a major difficulty when approaching these problems with population-based heuristics. However, evolutionary agent-based systems (EMAS) turn out to reduce the fitness function calls, which makes them a possible weapon of choice against them. This paper recalls the basics of EMAS and describes the considered problem (Step and Flash Imprint Lithography), and later, shows convincing results that EMAS is more effective than a classical evolutionary algorithm.*

Keywords multi-agent systems, evolutionary computation, inverse problems

1. Introduction

During the past decades, intelligent/autonomous software agents have been gaining more and more applications in various domains. Considered multi-agent systems (MAS) are based on intelligent interactions between the agents, such as coordination, cooperation or negotiation. Therefore, multi-agent systems are ideally suited to represent problems that have multiple problem-solving methods, multiple perspectives and/or multiple problem solving entities [27]. Apparently, agents play a key role in the integration of AI sub-disciplines, which often leads to hybrid design of modern intelligent systems.

Despite their high complexity, such systems are weapons of choice when dealing with difficult optimization problems. Inverse problems, belong to a wider class of so-called "black-box" problems consisting in finding optima of the function described in a space that is difficult to analyse using classical mathematical apparatuses are examples of such problems.

Solving such problems usually requires simulation based on a certain model and prediction of the behavior of the actual system based on its outcome. This classical approach is called a "forward problem"; however, "inverse problem" consists in influencing the model by feeding it with different parameters, usually coming from observing the actual phenomenon. Inverse problems are usually difficult to solve because different values of the model parameters may not be consistent with the data, and discovering the values of the model parameters may require the exploration of a huge parameter space.

The article concerns a hybrid evolutionary-agent approach. In most of the applications reported in literature (see e.g. [23] or [12] for a review), an evolutionary algorithm is used by an agent to aid realization of some of its tasks, often connected with learning or reasoning or supporting coordination of some group (team) activity. In other approaches, agents constitute a management infrastructure for a distributed realization of an evolutionary algorithm [25]. Yet, evolutionary processes are decentralized by nature, and indeed, one may imagine the incorporation of evolutionary processes into a multi-agent system at a population level. It means that, apart from interaction mechanisms typical of MAS (such as communication), agents are able to *reproduce* (generate new agents) and may *die* (be eliminated from the system). A similar idea, but with limited autonomy of agents located in fixed positions on some lattice (like in a cellular model of parallel evolutionary algorithms), was developed by e.g. [28].

The key idea of the decentralized model of evolution employed by an *evolutionary multi-agent system* – EMAS was proposed by Cetnarowicz in [11], and since then, it has been applied to different optimization problems (e.g., single-criteria, multi-criteria, discrete, continuous) [8]. The motivation for testing the EMAS algorithm on the linear elasticity with thermal expansion coefficient modelling, the step and flash imprint lithography (SFIL) problem concerns one of the most crucial features of the

inverse problem-solving with a metaheuristic approach: the fitness function is very complex and time-consuming.

EMAS and its variants (e.g., immunological one: iEMAS) have already proven to be much more effective than classical, parallel evolutionary algorithm in the means of fitness function calls (see, e.g., [4, 3, 5]). It is obvious that utilizing dedicated techniques aimed at reducing the number of processed individuals in the population may hamper exploration possibilities. However, the experimental results of EMAS and iEMAS yielded far better results than a parallel evolutionary algorithm (Michalewicz version [18]), especially in high-dimensional problems. The optima were also visibly approached faster (when considering the number of fitness function calls).

EMAS-like techniques may also be accused of being too complex to solve anything. Besides experimental evidence, an extensive formal study was conducted, and features of a dedicated, Markov-chain based model were analysed [9, 24, 7]. The research yielded that Markov-chain describing the dynamics of EMAS is ergodic, therefore EMAS may be counted to so-called Las Vegas algorithms (see [1]), having an asymptotic guarantee of success [22]. A similar proof has also been outlined for iEMAS [6].

Summing up these two highlights of EMAS-related research, the application of EMAS to inverse problems seems to be well-justified, both by existing analytical and experimental backgrounds.

In this paper, first the basics of evolutionary and agent-based computation and presentation of the concepts of the examined systems are recalled. Then, the inverse problem consisting in finding non-uniform Young moduli of the Step and Flash Imprint Lithography (SFIL) feature is described. The goal of the inverse analysis is to localize these 27 Young moduli, resulting in the measured deformation of the feature. The Finite Element-Method solver is utilized as a black-box, serving as a means for computing fitness function for EMAS (evolutionary multi-agent system [10]) and EA (evolutionary algorithm [18]) heuristics. Selected optimization results presented at the end of the paper encourage further research concerning such an application of agent-based computing, as better results are obtained after a lower number of costly fitness function calls, as in the case of EA.

2. Step and Flash Imprint Lithography

Step and Flash Imprint Lithography (SFIL) constitutes an important patterning framework used in silicon industry [13, 20]. The process consists of the following phases:

- dispense – depositing a low viscosity silicon containing photocurable etch barrier onto a substrate,
- imprint – bringing the template into contact with the etch barrier,
- expose – exposing the etch barrier to UV in order to cure it,
- separate – releasing the template.

Photopolymerization, however, is often accompanied by densification. The shrinkage of the feature can be modelled by linear elasticity with thermal expansion coefficient.

2.1. Linear elasticity model with thermal expansion coefficient

Following [15] the strong and weak formulations for the linear elasticity problem with thermal expansion coefficient are given as follows. The computational domain Ω is defined in the following way

$$\Omega = \{(x_1, x_2, x_3) : x_i \in (0, 1)\} \tag{1}$$

The bottom of the Ω constitute the Dirichlet boundary

$$\Gamma_D = \{(x_1, x_2, x_3) : x_1, x_2 \in (0, 1), x_3 = 0\} \tag{2}$$

and the remaining parts of the boundary of Ω constitute the Neumann boundary

$$\Gamma_N = \partial\Omega - \Gamma_D \tag{3}$$

Strong formulation. Given $g_i : \Gamma_D \ni \mathbf{x} \to g_i(\mathbf{x}) \in R$, θ and α_{kl}, find the displacement vector field $u_i : \bar{\Omega} \ni \mathbf{x} \to u_i(\mathbf{x}) \in R$, $i = 1, 2, 3$, such that

$$\sigma_{ij,j} = 0 \text{ in } \Omega, \tag{4}$$

$$u_i = g_i \text{ on } \Gamma_D, \tag{5}$$

$$\sigma_{ij} n_j = 0 \text{ on } \Gamma_N, \tag{6}$$

where σ_{ij} is the stress tensor, defined in terms of the generalized Hook's law

$$\sigma_{ij} = c_{ijkl}\left(\epsilon_{kl} + \theta\alpha_{kl}\right), \tag{7}$$

here c_{ijkl} are elastic coefficients (known for given material), θ is the temperature, α_{kl} are the thermal expansion coefficients, and $\epsilon_{ij} = u_{(i,j)} = \frac{u_{i,j}+u_{j,i}}{2}$ is the strain tensor, where $u_{i,j}$ are displacement gradients.

Weak formulation. The weak formulation is obtained by multiplying (4) by test functions $w_i \in H_0^1(\Omega)$ and integrating by parts over Ω:

$$-\int_\Omega w_{i,j}\sigma_{ij}d\Omega + \int_\Gamma w_i\sigma_{ij}n_j d\Omega = 0. \tag{8}$$

Since σ_{ij} is symmetric tensor, then $w_{i,j}\sigma_{ij} = w_{(i,j)}\sigma_{ij}$ and

$$\int_\Omega w_{(i,j)}\sigma_{ij}d\Omega = 0. \tag{9}$$

where we have also used the fact that $w_i = 0$ on Γ_D and $\sigma_{ij}n_j = 0$ on Γ_N. Finally, by utilizing (7) we get

$$\int_\Omega w_{(i,j)}c_{ijkl}u_{(k,l)}d\Omega = -\theta\int_\Omega w_{(i,j)}c_{ijkl}\alpha_{kl}d\Omega. \tag{10}$$

Reformulation for the SFIL modelling. For the convenient implementation of the algorithm, we utilize the following equivalent weak formulation. Find $\mathbf{u} \in \mathbf{V}$, such that

$$\mathbf{a}(\mathbf{u}, \mathbf{w}) = -\mathbf{A}(\mathbf{w}) \, \forall \mathbf{w} \in \mathbf{V}, \tag{11}$$

$$\mathbf{a}(\mathbf{u}, \mathbf{w}) = \int_{\Omega} \epsilon(\mathbf{w})^{\mathbf{T}} \mathbf{D} \epsilon(\mathbf{u}) \, \mathbf{d\Omega} \tag{12}$$

$$\mathbf{A}(\mathbf{w}) = \theta \int_{\Omega} \epsilon(\mathbf{w})^{\mathbf{T}} \mathbf{D} \alpha \mathbf{d\Omega} \tag{13}$$

where $= \{\mathbf{V} \in \left(H^1(\Omega)\right)^3 : tr\mathbf{v} = \mathbf{0} \text{on} \Gamma_D\}$, and Γ_D is defined as the bottom of the 3D cube. Here

$$\epsilon(\mathbf{u}) = \begin{pmatrix} u_{1,1} \\ u_{2,2} \\ u_{3,3} \\ u_{2,3} + u_{3,2} \\ u_{1,3} + u_{3,1} \\ u_{1,2} + u_{2,1} \end{pmatrix}, \tag{14}$$

$$\mathbf{D} = \frac{E}{(1+\nu)(1-2\nu)} \begin{pmatrix} 1-\nu & \nu & \nu & 0 & 0 & 0 \\ \nu & 1-\nu & \nu & 0 & 0 & 0 \\ \nu & \nu & 1-\nu & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1-2\nu}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1-2\nu}{2} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1-2\nu}{2} \end{pmatrix}. \tag{15}$$

2.2. Inverse problem

In order to calibrate the direct problem model, we need to find out the model parameters which involves the Young modulus, Poisson ratio and thermal expansion coefficient.

Following [13] we set up the Poisson ratio $\nu = 0.3$. We also assume that $g_i : \Gamma_D \ni \mathbf{x} \to g_i(\mathbf{x}) = 0$ (the feature is fixed at the bottom, with free boundary conditions on all other sides), $\theta = 1$ (the thermal expansion coefficient α expresses the volumetric contraction of the feature when the temperature gradient is equal to 1 Celsius), $\alpha_{ij} = -\alpha\delta_{ij}$ where $\alpha = -0.0615$ is based on inverse analysis [21].

In this paper, we want to find out the non-uniform Young modulus of the feature, resulting in slight lean of the feature, presented in Figure 1. We assume that there are 27 Young moduli for each of 27 sub-parts of the feature, summarized in Figure 2. The goal of the inverse analysis is to localize these 27 Young moduli, resulting in the measured deformation of the feature.

For the direct problem solution, we have utilized the self-adaptive hp finite element method application hp3d [19, 14], implementing the linear elasticity with thermal expansion coefficient.

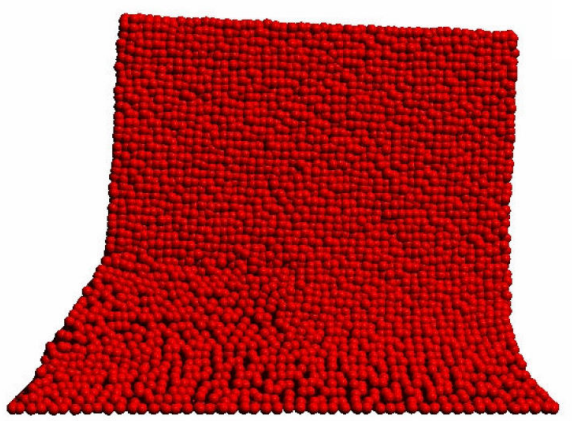


Figure 1. Slight lean of the feature.

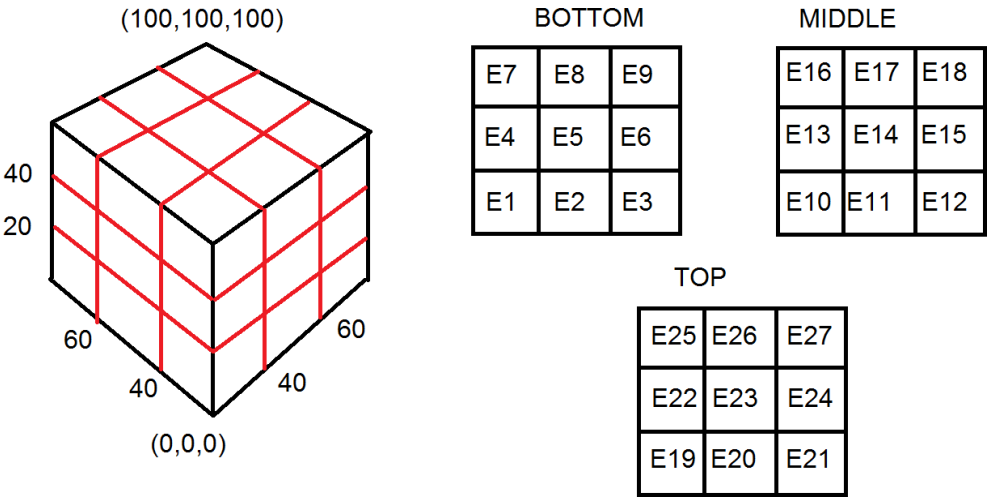


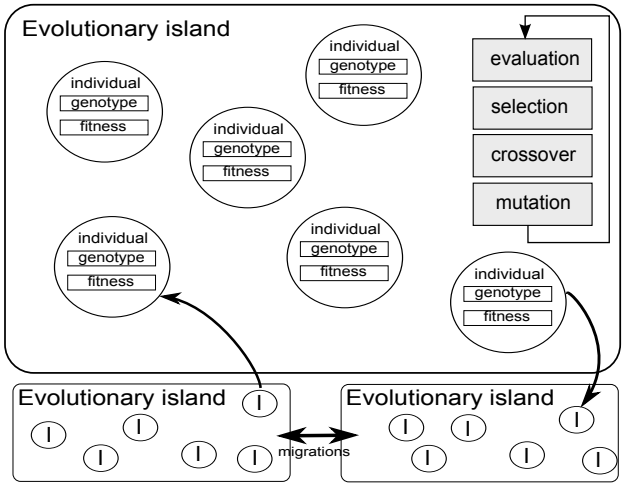
Figure 2. The problem considered.

3. Evolutionary multi agent computing

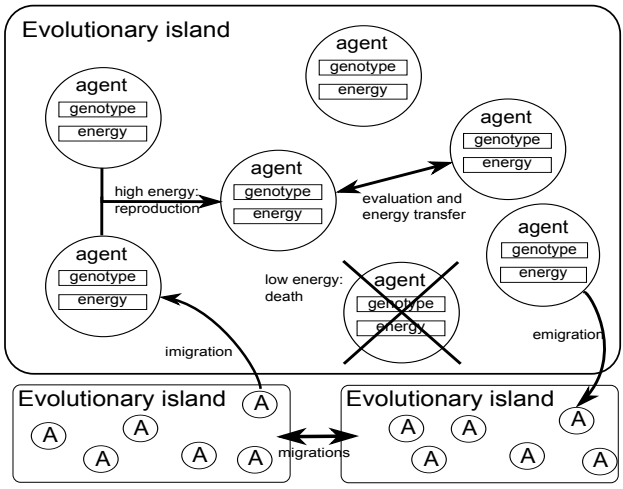
Agents of EMAS represent or generate solutions for a given optimization problem. They are located on islands which constitute their local environment where direct interactions may take place, and represent a distributed structure of computation. Obviously, agents are able to change their location, which allows for diffusion of information and resources all over the system [16].

In EMAS, phenomena of inheritance and selection–the main components of evolutionary processes–are modelled via agent actions of *death* and *reproduction* (see Fig.

3b). Inheritance is accomplished by an appropriate definition of reproduction, like in classical evolutionary algorithms. Core properties of the agent are encoded in its genotype and inherited from its parent(s) with the use of variation operators (mutation and recombination). Besides, an agent may possess some knowledge acquired during its lifetime which is not inherited. Both inherited and acquired information determine the behavior of an agent in the system (phenotype).



(a) Schematic presentation of an evolutionary algorithm



(b) Schematic presentation of an evolutionary multi-agent system

Figure 3. EA and EMAS outlines.

Assuming that no global knowledge is available (which makes it impossible to evaluate all individuals at the same time) and autonomy of the agents (which causes reproduction to be achieved asynchronously), selection is based on the non-renewable resources [10]. Thus, a decisive factor of the agent's activity is its fitness, expressed by the amount of non-renewable resource it possesses. The agent gains resources as a reward for 'good' behavior, and loses resources as a consequence of 'bad' behavior. Selection is realized in such a way that agents with a lot of resources are more likely to reproduce, while low energy increases the possibility of death.

The main advantage of the approach is the coverage of various specialized evolutionary techniques in one coherent model. Concerning computational systems, EMAS enables the following:

- local selection allows for intensive exploration of the search space, which is similar to parallel evolutionary algorithms,
- the way phenotype (behavior of the agent) is developed from genotype (inherited information) depends on its interaction with the environment,
- self-adaptation of the population size is possible when appropriate selection mechanisms are used.

What is more, explicitly-defined living space facilitates implementation in a distributed computational environment.

Solving optimization problems with evolutionary algorithms requires that the following must be defined [2]: appropriate encoding of the solutions, crossover and mutation operators appropriate for the encoding, choosing a selection mechanism, and possibly other components of specialized techniques like configuring topology of islands and migration strategies for the island model of parallel evolutionary algorithms.

In the simplest possible model of an evolutionary multi-agent system, there is one type of agent and one resource defined. Genotypes of agents represent feasible solutions to the problem.

Energy is exchanged by agents in the process of evaluation. The agent increases its energy when it finds out that one (e.g. randomly chosen) of its neighbors has lower fitness. In this case, the agent takes a part of its neighbor's energy; otherwise, it passes part of its own energy to the evaluated neighbor. The level of life energy triggers actions of death and reproduction (low energy causes death while high energy makes reproduction possible).

EMAS agents may perform the following actions:

- *Reproduction* – performed when the agent's energy raises above a certain level, followed by production of a new individual in cooperation with one of its neighbors, with genotype based on parents' genotypes (crossed over and mutated) and part of energy (usually half of its initial value) also passed from each of its parents.
- *Death* – agent is removed from the system when its energy falls below a certain level, the remaining energy is distributed among its neighbors.

- *Evaluation* – agent chooses its neighbor and compares the fitness of its genotype with its own; in the case when the neighbor is better, it receives part of the agent's energy, and vice versa.
- *Migration* – agent (with some probability) may migrate, then it is removed from one evolutionary island and moved to another (random) according to predefined topology.

Each action is attempted randomly with certain probability, and it is performed only when basic preconditions are met (e.g. an agent may attempt to perform the action of reproduction, but it will reproduce only if its energy rises above certain level and it meets an appropriate neighbor).

The topology of an island defining the structure of inter-agent relations may be random (full graph of connections between the agents); but in order to enhance diversity of the population, an additional level of population decomposition besides the evolutionary islands) may be introduced. Thus, a two-dimensional square lattice (similarly to the ones used in Cellular Automata [26]) may be considered. In such a lattice, different neighborhoods (e.g., Moore's) and boundary conditions (e.g., periodic, reflexive and fixed) may be utilized.

In such an island, the agents may interact between themselves, provided they are in the zone of each other's neighborhood.

4. Experimental results

Having experience in the development of component-based agent-oriented computing platforms (cf. AgE[1]), a simplified version of such a discrete-event simulation and computing system was developed using Python technology. The choice of this technology was undertaken based on a relatively easy implementation process and high portability [17]. Using this software environment, both EMAS and EA systems were implemented and used to generate the presented results. All possible parameters of the both systems were set to the same value. The configuration of the both systems is presented as follows. The exact values of these parameters have been based on the results already presented in [5] and tuned up in trial-and-error process.

Common parameters:

- Mutation: continuous distribution-based modification of one randomly chosen gene.
- Crossover: single-point, the descendant gets parts of its parents genotype after dividing them in one randomly chosen point.
- Problem: 27-dimensional described earlier problem with error (precision) 60% (30 repetitions of experiment), 25% (5 repetitions of experiment) and 8% (single run of experiment). The dispersion of results was pointed out in graphs using error bars (based on standard deviation).

[1] http://age.iisg.agh.edu.pl

- Values boundaries for each Young modulus: $[10^5, 10^{10}]$.
- Agent/individual mutation probability: 0.2.
- Population size: 15.

EA parameters:

- Number of steps: 1000.
- Mating pool size equals to number of individuals.

EMAS parameters:

- Number of steps: 10000.
- Initial energy: 100, received by the agents in the beginning of their lives.
- Minimal reproduction energy: 90, required to reproduce.
- Evaluation energy win/loose: 20/-20, passed from the looser to the winner.
- Death energy level: 0, used to decide which agent should be removed from the system.
- Intra-island neighborhood: Moore's, each agent's neighborhood consists of 8 surrounding cells.
- Size of 2-dimensional lattice as an environment: 10×10.

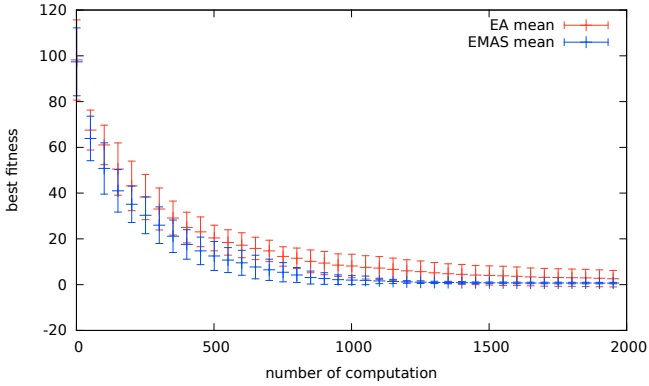
A number of steps was arbitrary chosen to satisfy final average fitness value on level about 1.0. This parameter differs for EMAS and EA for one more reason: the most important observations (e.g., best fitness) were noted in relation to the number of fitness computation (instead of subsequent step of computation or arbitrarily measured time).

Initial values of genes were randomly generated from continuous space defined by boundaries. They represent 27 Young moduli and, with precision, are input to hp3d application which outputs deformation values.

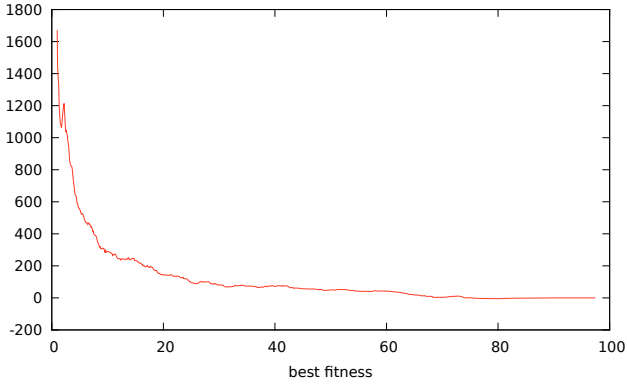
The outputs from the hp3d code are the minimum and maximum displacements of the feature along x, y and z axes. Fitness function is calculated using mean squared error between output values of the minimum / maximum displacements obtained from hp3d code and the minimum / maximum displacements obtained from experimentation.

Experiments were conducted with three different precisions. Time of fitness function evaluation depends on this parameter as presented in Table 1. Experiments were repeated to ensure independence from initial values. The numbers of experiments repetitions were chosen to finish them in reasonable time as shown in Table 2, therefore the most reliable results from the statistical point of view are shown for the precision 60%.

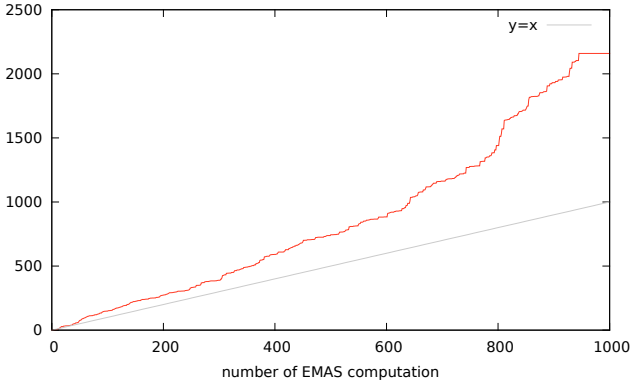
All the conducted experiments revealed that EMAS produced better results far earlier than EA (see Figs. 4, 5, 6). These differences were visualized in Figs. 6b, 5b, 4b. Only for the precision 25%, EA reached the same fitness level as EMAS; however, not earlier than after computing 800th fitness function call.



(a) Average best fitness of EA and EMAS

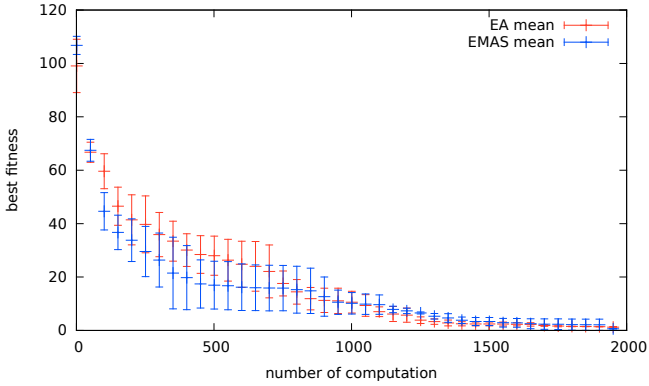


(b) Difference between numbers of fitness function evaluations needed to achieve equal fitness value by EA and EMAS

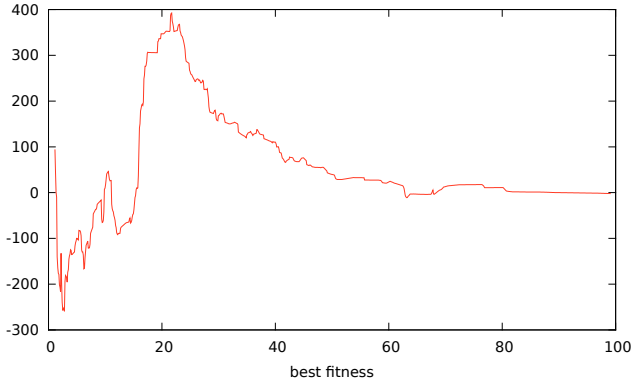


(c) Number of EA fitness computation where it obtains the same fitness as EMAS

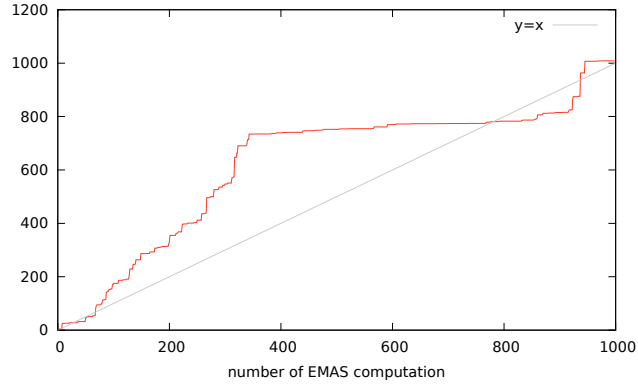
Figure 4. Results of the experiments computed with error 60%.



(a) Average best fitness of EA and EMAS

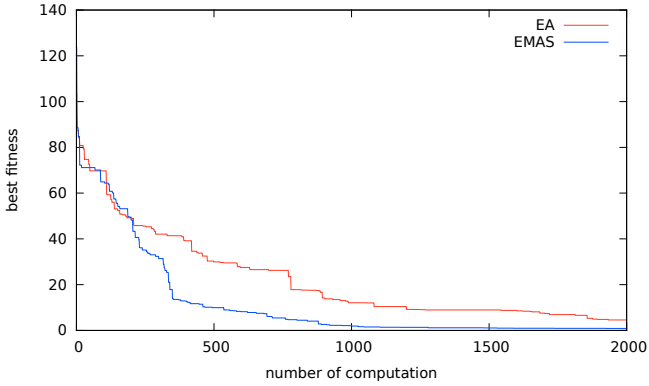


(b) Difference between numbers of fitness value computation by EA and EMAS

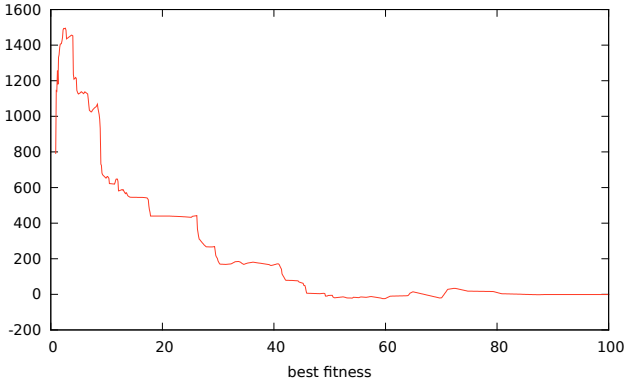


(c) Number of EA fitness computation where it obtains the same fitness as EMAS

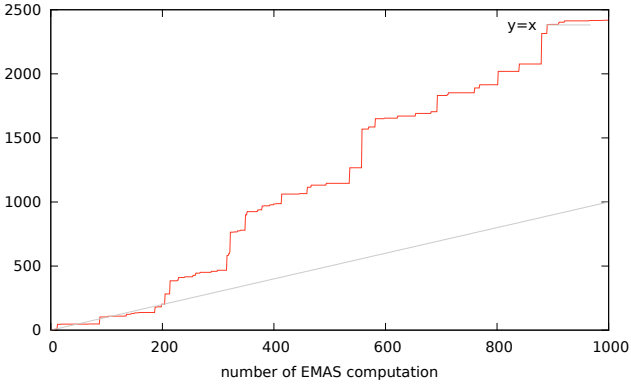
Figure 5. Results of the experiments computed with error 25%.



(a) Best fitness of EA and EMAS



(b) Difference between numbers of fitness value computation by EA and EMAS



(c) Number of EA fitness computation where it obtains the same fitness as EMAS

Figure 6. Results of the experiments computed with error 8%.

Table 1

Execution times of fitness function evaluations on different processors with different precisions.

Precision	Intel i7 2670QM 2.2GHz	AMD Opteron 1220 2.8GHz
100–60	0m0.533s	0m1.053s
59–25	0m53.234s	2m37.105s
24–8	2m31.210s	7m41.221s
7–5	4m11.729s	12m49.887s
4–1	6m26.175s	19m7.572s

Table 2

Execution time of experiments on AMD Opteron 1220 2.8GHz.

Precision	Repetitions	EMAS	EA
60	30	17h30m	26h20m
25	3	10d22h	16d8h
8	1	10d16h	16d

Found solution on acceptable precision (8%) has fitness value 0.876222 and 27 Young moduli are: 1.9e+09, 5.1e+08, 1.6e+09, 3.4e+08, 1.4e+08, 5.1e+08, 1.2e+08, 1.1e+09, 1.1e+09, 1.2e+09, 4.9e+08, 6.5e+08, 1.7e+09, 1.4e+09, 3.7e+07, 5.2e+08, 9.1e+08, 3.8e+08, 6.6e+07, 1.3e+09, 4.8e+08, 8.7e+08, 1.8e+09, 2.9e+08, 1.0e+09, 7.4e+08, 2.4e+08.

Figure 7. Visualization of found solution. Solution with fitness 0.876222 with precision 8

5. Conclusions

Inverse problems pose a challenging task for solving, especially using population-based meta-heuristics, because of a significant burden: a very costly fitness function call. Therefore, approaches reducing the number of fitness function may become

very valuable. In this paper, the approach to solve such problems using evolutionary agent-based systems (EMAS) was presented. The obtained results clearly showed that this approach was successful in the considered problems, namely EMAS, turned out to be more effective than EA. The experiments were repeated, and the average with standard deviation revealed that these experiments were repeatable. The most reliable experiments were conducted for quite a low precision (60%), as the runtime of experiments was barely acceptable. In the future, the authors plan to repeat the experiments with a wider range of parameters; also, additional inverse problems will be approached with this promising agent-based computing methodology.

Acknowledgements

The work presented in this paper was partially supported by the Polish National Science Centre research projects No. DEC-2011/03/B/ST6/01393 "Multi-model, multi-criterial and multi-adaptive strategies for solving the inverse tasks", and No. N N516 500039 "Biologically inspired mechanisms in planning and management of dynamic environments".

References

[1] Atallah M.: *Algorithms and Theory of Computation Handbook*. CRC Press LLC, 1999.

[2] Bäck T., Fogel D., Michalewicz Z., editors. *Handbook of Evolutionary Computation*. IOP Publishing and Oxford University Press, 1997.

[3] Byrski A., Kisiel-Dorohinicki M.: Immunological selection mechanism in agent-based evolutionary computation. In Klopotek M., Wierzchon S., Trojanowski K., editors, *Proc. of the Intelligent Information Processing and Web Mining IIS IIPWM '05: Gdansk, Poland*, Advances in Soft Computing. Springer Verlag, 2005.

[4] Byrski A., Kisiel-Dorohinicki M.: Agent-based evolutionary and immunological optimization. In *Computational Science – ICCS 2007, 7th International Conference, Beijing, China, May 27–30, 2007, Proceedings*. Springer, 2007.

[5] Byrski A., Korczyński W., Kisiel-Dorohinicki M.: Memetic multi-agent computing in difficult continuous optimisation. In *Proc. of 6th International KES Conference on Agents and Multi-agent Systems Technologies and Applications, 2013, Hue City, Vietnam, IOS Press (accepted in 2013)*. Springer.

[6] Byrski A., Schaefer R., Smołka M.: Markov chain based analysis of agent-based immunological system. *Transactions on Computational Collective Intelligence*, 10:1–15, 2013.

[7] Byrski A., Schaefer R., Smołka M.: Asymptotic guarantee of success for multi-agent memetic systems. *Bulletin of the Polish Academy of Sciences–Technical Sciences*, 61(1), 2013.

[8] Byrski A., Dreżewski R., Siwik L., Kisiel-Dorohinicki M.: Evolutionary multi-agent systems. *The Knowledge Engineering Review*, Accepted for publication, 2012.

[9] Byrski A., Schaefer R.: Stochastic model of evolutionary and immunological multi-agent systems: Mutually exclusive actions. *Fundamenta Informaticae*, 95(2–3): 263–285, 2009.

[10] Cetnarowicz K., Kisiel-Dorohinicki M., Nawarecki E.: The application of evolution process in multi-agent world (MAW) to the prediction system. In Tokoro M., editor, *Proc. of the 2nd Int. Conf. on Multi-Agent Systems (ICMAS'96)*. AAAI Press, 1996.

[11] Cetnarowicz K.: Evolution in multi-agent world = genetic algorithms + aggregation + escape. In *7th European Workshop on Modelling Autonomous Agents in a Multi-Agent World (MAAMAW'96)*. Vrije Universiteit Brussel, Artificial Intelligence Laboratory, 1996.

[12] Chen S.-H., Kambayashi Y., Sato H.: *Multi-Agent Applications with Evolutionary Computation and Biologically Inspired Technologies*. IGI Global, 2011.

[13] Colburn M.: *Step and Flash Imprint Lithograpy: A Low Pressure, Room Temperature Nonoimprint Lithography*. PhD thesis, The University of Texas in Austin, 2000.

[14] Demkowicz L., Kurtz J., Pardo D., Paszyński M., Rachowicz W., Zdunek A.: Computing with hp-adaptive finite elements. *Frontiers: Three Dimensional Elliptic and Maxwell Problems with Applications*. Chapman & Hall/CRC, 2007.

[15] Hughes T.: *The Finite Element Method. Linear Statics and Dynamics Finite Element Method Analysis*. Dover, 2000.

[16] Kisiel-Dorohinicki M.: Agent-oriented model of simulated evolution. In Grosky W., Plil F., editors, *SOFSEM 2002: Theory and Practice of Informatics*, vol. 2540 of *Lecture Notes in Computer Science*, pp. 253–261. Springer, Berlin Heidelberg, 2002.

[17] Lutz M.: *Programming Python*. O'Reilly Media, 2011.

[18] Michalewicz Z.: *Genetic Algorithms Plus Data Structures Equals Evolution Programs*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1994.

[19] Paszyński M., Demkowicz L.: Parallel, fully automatic hp-adaptive 3d finite element package. *Engineering with Computers*, 22(3):255–276, 2006.

[20] Paszyński M., Romkes A., Collister E., Meiring J., Demkowicz L., Willson C.: On the modeling of step and flash imprint lithography. Technical report, ICES Report 05-38, 2005.

[21] Paszyski M., Barabasz B., Schaefer R.: Efficient adaptive strategy for solving inverse problems. In Shi Y., Albada G., Dongarra J., Sloot P., editors, *Computational Science ICCS 2007*, vol. 4487 of *Lecture Notes in Computer Science*, pp. 342–349. Springer, Berlin Heidelberg, 2007.

[22] Rinnoy Kan A., Timmer G.: Stochastic global optimization methods. *Mathematical Programming*, 39:27–56, 1987.

[23] Sarker R., Ray T.: *Agent-Based Evolutionary Search*. Springer, 2010.

[24] Schaefer R., Byrski A., Smołka M.: Stochastic model of evolutionary and immunological multi-agent systems: Parallel execution of local actions. *Fundamenta Informaticae*, 95(2–3):325–348, 2009.

[25] Schaefer R., Kołodziej J.: Genetic search reinforced by the population hierarchy. *Foundations of Genetic Algorithms*, 7, 2003.

[26] Wolfram S.: *A New Kind of Science*. Wolfram Media, 2002.

[27] Wooldridge M.: *An Introduction to Multiagent Systems*. John Wiley & Sons, 2009.

[28] Zhong W., Liu J., Xue M., Jiao L.: A multiagent genetic algorithm for global numerical optimization. *IEEE Trans. on Systems, Man, and Cybernetics, Part B: Cybernetics*, 34(2):1128–1141, 2004.

Affiliations

Krzysztof Wróbel
AGH University of Science and Technology, Krakow, Poland, wrkrzysz@student.agh.edu.pl

Paweł Torba
AGH University of Science and Technology, Krakow, Poland, tpawel@student.agh.edu.pl

Maciej Paszyński
AGH University of Science and Technology, Krakow, Poland, paszynsk@agh.edu.pl

Aleksander Byrski
AGH University of Science and Technology, Krakow, Poland, olekb@agh.edu.pl

Received: 13.01.2013
Revised: 18.02.2013
Accepted: 21.03.2013